FE project :

THE DEVELOPMENT CHRONICLE

The red box

Index

The redbox

https://github.com/js-HARDESTGAME/HARDESTGAME.git

https://github.com/junwon-hwang

황준원

https://github.com/Maybaba

신윤종

https://github.com/kimbyeongmo

김병모

https://github.com/hgb926

한기범

background

- 어렸을 때부터 자주 접했던 아케이드 장르
- 대다수의 사람들이 플레이해보았던 게임
- 게임 내 오브젝트의 간단한 함수 구현
- 복잡하지 않고 직관적인 UI

승부욕을 자극하는
게임이야!

한번 하면 승부욕에
계속하게 돼

간단하고 구현하기
수월해보인다

어렸을 때 엄청
유행했던 게임이야!

다같이 재미있게
즐길 수 있는 게임!

you튜브에서
게임방송하는거
봤어.

언제나 썰리는건
나였다

그거 나도
해본 게임이야!

As is / To be

- 다양한 맵과 이벤트 등의 업데이트 부재
- 캐릭터의 반응속도 느림 -> 지루한 느낌

- 새로운 맵 추가
- 기존게임보다 반응속도 증가
- 캐릭터 움직임 부드럽게 구현
- 메인 페이지 디자인 업데이트
- 프로젝트 목표처럼 장애물 추가, 텔레포트 등의 역동적인 기능 업데이트

Benefits / UX benefits

- 역동적인 기능 추가로 잠재고객의 유입 촉진

- 고객이 직접 맵을 구성하여 게임에 추가할 수 있음으로, 고객들의 참여와 장기적인 이용을 유도

- 맵 제작 유료화와 기능 추가 유료화를 통해 수익 창출

- 극복감과 자기만족감

- 집중력 향상 및 반응속도 증가

- 인내와 문제해결능력 향상

Schedule

20240404 (목)

회의 및
개발 기능 분담

20240405 (금)

각자 맵
css html으로
구현

240408 (월) ~ 10(수)

기능 개발

240411 (목) ~ 12(금)

기능병합 및
버그픽스

240415(월)

프로젝트 발표

Tech Stacks

Team introduction

공통 : 각자 맵 디자인, import export 및 git 활용한 유지보수

- 노란공 충돌 함수 구현
- 파란공 충돌시
  노란공 리셋 구현

황준원

- 파란 공 충돌 함수
- 파란 공 충돌 시
  애니메이션 구현
- 게임메인페이지 및 파비콘 제작

신윤종

- 빨간 박스 이동 함수
- 영역 내에서만 이동
- 자연스러운 움직임 구현

김병모

- 클리어 함수 구현
- 페이지 간 링크 연결
- 상 하 배너 디자인
- 히든 클리어 제작

한기범

황준원

Troubleshooting

1. 노란 공과 빨간 박스 충돌시 노란 공의 디스플레이 값을 NONE으로 수정하는 방법으로 코드 구상.

2. 충돌을 확인하기 위한 방법을 고민하던 중 getBoundingClientRect() 함수를 통하여 노란 공과 빨간 박스의 X좌표값을 확인 함.

3. 실행 시 최초에만 X좌표값을 확인하여 충돌 여부를 확인 할 수 없는 문제가 발생함.

4. setInterval()를 사용하여 X좌표의 값을 반복적 확인함.

5. 위 동일한 방법으로 노란 공과 파란 공 충돌시 노란공의 display값을 수정하는 방법으로 노란공 리셋기능 구현함.

```
// 충돌 함수
function isColliding(rect1, rect2) {
  return !(
    // 노란공과 빨간공의 좌표값을 비교하는 부분
    rect1.right < rect2.left ||
    rect1.left > rect2.right ||
    rect1.bottom < rect2.top ||
    rect1.top > rect2.bottom
  );
}

const intervalId = setInterval(function () {
  const redBoxRect = $redBox.getBoundingClientRect();
  let result = 0;

  Array.from($eatCircle1).forEach((circle) => {
    const circleRect = circle.getBoundingClientRect();
    if (isColliding(redBoxRect, circleRect)) {
      circle.style.display = "none";
    }
    if (circle.style.display !== "none") {
      result++;
    }
  });

//  반복문을 통하여 노란공의 디스플레이값을 블록으로 수정하여
  //  노란공을 재생성하는 함수
  Array.from($eatCircle1).forEach((circle) => {
    circle.style.display = "block";
  });
```

Team introduction

공통 : 각자 맵 디자인, import export 및 git 활용한 유지보수

- 노란공 충돌 함수 구현
- 파란공 충돌시
  노란공 리셋 구현

황준원

- 파란 공 충돌 함수
- 파란 공 충돌 시
  애니메이션 구현
- 게임메인페이지 및 파비콘 제작

신윤종

- 빨간 박스 이동 함수
- 영역 내에서만 이동
- 자연스러운 움직임 구현

김병모

- 클리어 함수 구현
- 페이지 간 링크 연결
- 상 하 배너 디자인
- 히든 클리어 제작

한기범

신윤종

Troubleshooting

1. 빨간박스가 파란 공(적)과 충돌하면 재시작 위치로 돌아오도록 코드 작성
2. 재시작 위치는 html로 잡은 초록색 칸 div의 아이디요소를 추출해 충돌 시 재시작 위치로 잡음
3. 원활하게 돌아감
4. 충돌 시 애니메이션 함수를 추가함
    a. 1초동안 해당 자리에서 멈추기
        i. 1초동안 충돌한 위치로 고정함
        ii. 충돌한 위치에 고정시켰으나
    b. 그러면서 1초동안 빨간박스의 opacity → 0으로 감
    c. opacity가 0으로 돌아갔을 때 재시작 위치로 위치 재설정하기
5. 이벤트를 실행시켰을 때 재시작 위치로 돌아가지 않았음
6. 기존 html과 css 위치를 가져와서 알고리즘 재구현
7. 충돌 내 알고리즘 활용한 단계별 함수 구현

a. 충돌 함수

```
function isColliding(rect1, rect2) {

  return !(
    rect1.right < rect2.left ||
    rect1.left > rect2.right ||
    rect1.bottom < rect2.top ||
    rect1.top > rect2.bottom
  );

}
```

b. 키보드 이벤트 활성화 된 경우에만 움직이는 함수

```
function moveBox() {
  if (canMove) { // 키보드 이벤트가 활성화된 경우에만 움직임
    if ("ArrowLeft" in keys && !checkCollision("left")) {
      x = Math.max(x - step, 0);
    }
    if ("ArrowRight" in keys && !checkCollision("right")) {
      x = Math.min(window.innerWidth - boxSize, x + step);
    }
    if ("ArrowUp" in keys && !checkCollision("up")) {
      y = Math.max(y - step, 0);
    }
    if ("ArrowDown" in keys && !checkCollision("down")) {
      y = Math.min(window.innerHeight - boxSize, y + step);
    }
  }
```

c. 빨간박스가 시작하는 위치를 재시작 위치로 설정하는 함수

```
function resetRedBoxPosition() {
  // 재시작 위치로 돌아가는 애니메이션 추가
  $redBox.style.transition = "none"; // 애니메이션 초기화
  $redBox.style.opacity = 1; // opacity 초기화

  x = initialX;
  y = initialY;
  drawBox();
}
```

Team introduction

공통 : 각자 맵 디자인, import export 및 git 활용한 유지보수

- 노란공 충돌 함수 구현
- 파란공 충돌시
  노란공 리셋 구현

황준원

- 파란 공 충돌 함수
- 파란 공 충돌 시
  애니메이션 구현
- 게임메인페이지 및 파비콘 제작

신윤종

- 빨간 박스 이동 함수
- 영역 내에서만 이동
- 자연스러운 움직임 구현

김병모

- 클리어 함수 구현
- 페이지 간 링크 연결
- 상 하 배너 디자인
- 히든 클리어 제작

한기범

김병모

Troubleshooting

1. 빨간 박스의 스타일, 초기 위치, 이동할 거리 설정 및 가져오기

2. 키보드 이벤트를 활용하여 빨간 박스 이동 구현
    - 2-1. 버퍼링 걸린 것처럼 움직임
    - 2-2. `requestAnimationFrame()`을 사용하여 브라우저에게 애니메이션을 요청하고, 각 프레임마다 빨간 상자의 위치를 업데이트하여 부드러운 움직임을 구현

3. 임의로 설정한 영역 내에서만 움직이게 하기
    - 3-1. 초기 구현 방식은 빨간 박스가 넘어가면 안 되는 부분에는 `display="none"` 값을 주어서 빨간 박스가 "none"으로 설정한 부분에 도달하면 키보드 이벤트를 꺼버리는 방식을 생각함
    - 3-2. 하지만 원하는 결과를 얻지 못함
    - 3-3. 맵의 모양을 잡기 위해 `border`처리한 부분 즉, 경계면의 모든 좌표값을 구한 후
    - 3-4. 빨간 박스와 경계면의 충돌 함수 생성
    - 3-5. 초기 충돌값을 `false`로 설정한 후, 이동 방향에 따라 충돌 검사를 함
    - 3-6. 충돌이 발생하면 충돌값을 `true`를 반환하게 함
    - 3-7. 빨간 박스 이동함수에 `if`문을 걸고 조건식에 방향키값이랑 충돌 검사를 하게 되면
    - 3-8. 임의 설정한 영역 내에서만 이동하기 구현 완료

김병모

Troubleshooting

```
const $boxStyle = getComputedStyle($redBox);
let x = parseInt($boxStyle.left);
let y = parseInt($boxStyle.top);
const initialX = x;
const initialY = y;
const boxSize = parseInt($boxStyle.width);
const step = 5;
```

```
function moveBox() {
  if (canMove) {
    // 키보드 이벤트가 활성화된 경우에만 움직임
    if ("ArrowLeft" in keys && !checkCollision("left")) {
      x = Math.max(x - step, 0);
    }
    if ("ArrowRight" in keys && !checkCollision("right")) {
      x = Math.min(window.innerWidth - boxSize, x + step);
    }
    if ("ArrowUp" in keys && !checkCollision("up")) {
      y = Math.max(y - step, 0);
    }
    if ("ArrowDown" in keys && !checkCollision("down")) {
      y = Math.min(window.innerHeight - boxSize, y + step);
    }
  }

  drawBox();
  requestAnimationFrame(moveBox);
}

function drawBox() {
  $redBox.style.left = x + "px";
  $redBox.style.top = y + "px";
}
```

김병모

Troubleshooting

```
function checkCollision(direction) {
  const $boxRect = $redBox.getBoundingClientRect();
  const $obstacles = document.querySelectorAll(
    ".leftborder, .rightborder, .topborder, .bottomborder, .leftLine, .rightLine, .topLine, .bottomLine"
  );

  let collision = false;

  $obstacles.forEach(function ($obstacle) {
    const obstacleRect = $obstacle.getBoundingClientRect();

    switch (direction) {
      case "left":
        if (
          $boxRect.left - 10 < obstacleRect.right &&
          $boxRect.right > obstacleRect.right &&
          $boxRect.top < obstacleRect.bottom &&
          $boxRect.bottom > obstacleRect.top
        ) {
          collision = true;
        }
        break;
      case "up":
        if (
          $boxRect.top - 10 < obstacleRect.bottom &&
          $boxRect.bottom > obstacleRect.bottom &&
          $boxRect.left < obstacleRect.right &&
          $boxRect.right > obstacleRect.left
        ) {
          collision = true;
        }
        break;
      case "right":
        if (
          $boxRect.right + 10 > obstacleRect.left &&
          $boxRect.left < obstacleRect.left &&
          $boxRect.top < obstacleRect.bottom &&
          $boxRect.bottom > obstacleRect.top
        ) {
          collision = true;
        }
        break;
```

```
      case "down":
        if (
          $boxRect.bottom + 10 > obstacleRect.top &&
          $boxRect.top < obstacleRect.top &&
          $boxRect.left < obstacleRect.right &&
          $boxRect.right > obstacleRect.left
        ) {
          collision = true;
        }
        break;
    }
  });

  return collision;
}

moveBox();
```

Team introduction

공통 : 각자 맵 디자인, import export 및 git 활용한 유지보수

- 노란공 충돌 함수 구현
- 파란공 충돌시
  노란공 리셋 구현

황준원

- 파란 공 충돌 함수
- 파란 공 충돌 시
  애니메이션 구현
- 게임메인페이지 및 파비콘 제작

신윤종

- 빨간 박스 이동 함수
- 영역 내에서만 이동
- 자연스러운 움직임 구현

김병모

- 클리어 함수 구현
- 페이지 간 링크 연결
- 상 하 배너 디자인
- 히든 클리어 제작

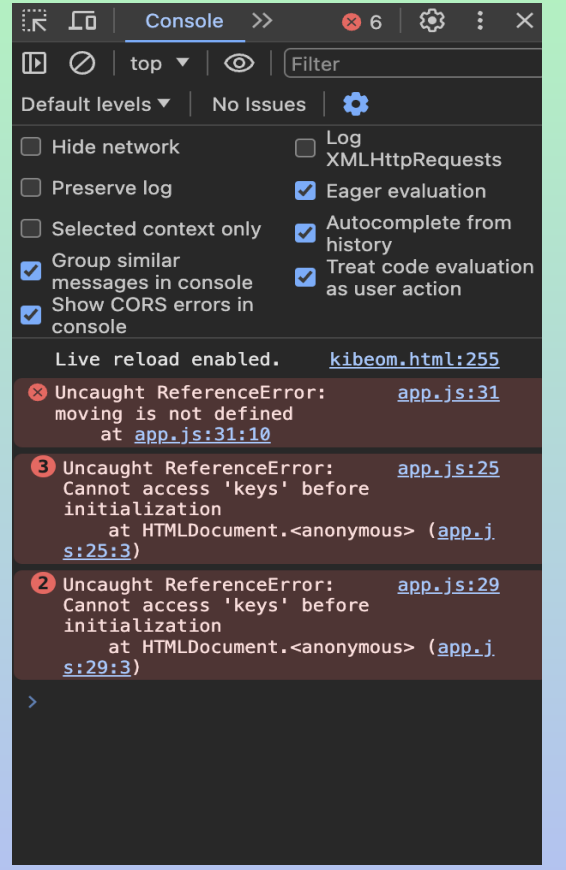
한기범

한기범

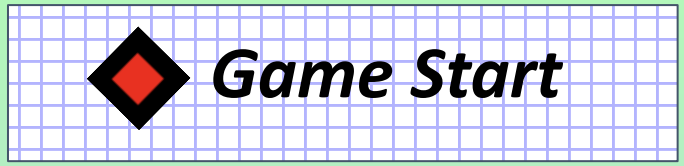
Troubleshooting

1. 함수를 만드는 과정보다, 함수들을 합치는 과정에서 겪은 에러들이 많았습니다.

2. 각각의 기능을 독립적으로 개발하고 병합한 후, 실행 시점에서 이전에는 발생하지 않았던 문제가 발생했습니다.
문제는 여러 함수가 결합되면서 호출되는 시점에 있었습니다.
이를 인지하고 문제를 해결하기 위해 코드를 재검토하고 수정했습니다.



?!



https://maybaba.github.io/

Future Update Planed

- 난이도 easy, normal, hard 추가 예정
- 닉네임, 타임어택, 순위 기능 추가
- 더욱 다양한 맵 제작
- BGM 추가

Project Learned

각자의 기능을 구현해 병합하는데 어려움이 있었고, 해결하는 과정에서 코드 이해 및 소통등 많은 배움이 있었다
개인사정으로 끝까지 마무리 못하여서 아쉬웠다.

황준원

충돌 함수를 구현할 때 처음에는 한 가지 로직만 고민하고 실행하려 했고, 문제가 발생하면 그 로직에서 해결하려고 애를 썼다. 하지만 결국 문제가 해결되지 않아 주변 사람들과 문제를 공유하고 조언을 구하였다. 팀원, 선생님, 그리고 인터넷을 통해 자신의 코드를 공유하고 의견을 듣는 것이 중요하다는 것을 깨달았다. 이를 통해 유연한 사고와 다양한 해결법을 받아들일 수 있었고, 문제를 수월하게 해결할 수 있었다.

신윤종

다양한 문제에 직면했지만 팀원들과 협업하고 때로는 외부의 도움을 받으면서 다른 시각에서 문제를 바라볼 수 있는 능동적인 태도가 중요하다는 것을 깨달았다.또한, 새로운 해결책을 찾는 과정에서 더 많은 것을 배우고 습득할 수 있다는 것도 알게 되었다.

김병모

하나의 게임을 여러명에서 작업하며 병합을 하다보면 깔끔한 코드의 내용을 유지하기 어렵다 느꼈고, 이를 통해 디버깅이 수월해지려면 회의에서 여러 약속을 더욱 탄탄히 정해야 겠다 느꼈다.

김기범